LISTING OF CLAIMS:

Claims 1-8 (Cancelled).

Claim 9: (Previously Presented) An optimization method for optimizing a program to increase processing efficiency on a target machine, comprising the steps of:

establishing a basic block that includes a Try node for examining, in an object program, a command that may cause an exception process on the target machine to determine whether an exception process has occurred, and a Catch node for performing an inherent process when an exception process has occurred on the target machine;

performing optimization of said object program wherein said basic block is established;

examining said optimized object program to determine whether a difference in content exists between said command that may cause an exception process on the target machine and an exception handler whereat said exception process is performed; and

generating in said Catch node of said basic block when a difference exists, a compensation code for compensating for said difference, and a code for, after said compensation code has been obtained, moving program control to said exception handler whereat said exception process is performed; and

wherein said step of examining said object program to determine whether a difference in content exists between said command that may cause an exception process and said exception handler whereat said exception process is performed includes a step of:

removing said basic block prepared for said command that may cause an exception process when no difference in content exists.

Claim 10 (Cancelled).

Claim 11: (Previously Presented) An optimization method for optimizing a program to increase processing efficiency on a target machine, comprising the steps of:

dividing software code, in an object program, that may cause an exception process on the target machine into software code for determining whether an exception process has occurred and software code for actually causing an exception process;

specifying said code obtained at said division step as branches of a control flow graph;

designing said control flow graph so that when an exception process occurs on the target machine, program control is shifted to said code that actually caused said exception process;

performing said optimization process for said object program that has been modified;

determining whether code for compensating for a difference in content between the point of origin of an exception process and code for actually causing said exception process have been generated in a block that includes code for the actual performance of said exception process after the optimization process has been run; and

using said control flow graphs, when said code for compensating for said content difference is not generated, for synthesizing said two code sets to obtain said code arrangement that existed before said code was divided.

Claims 12-18 (Cancelled).